OSCAR IPT/Bold Stroke
Open Systems Lessons Learned

Prepared by the OSCAR IPT for:

Glenn T. Logan - Lt Col USAF

Open Systems Joint Task Force

REPORT DOCUMENTATION PAGE

Form Approved OMB No. 0704-0188

1. REPORT DATE (DD-MM-YYYY) 26-02-2000	2. REPORT TYPE Briefing	3. DATES COVERED (FROM - TO) xx-xx-2000 to xx-xx-2000

4. TITLE AND SUBTITLE OSCAR IPT/Bold Stroke Open Systems Lessons Learned Unclassified	5a. CONTRACT NUMBER 5b. GRANT NUMBER 5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S) Logan, Glenn T. ;	5d. PROJECT NUMBER 5e. TASK NUMBER 5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME AND ADDRESS Open Systems Joint Task Force 2001 N. Beauregard St., Suite 800 Alexandria, VA22311	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME AND ADDRESS Open Systems Joint Task Force (OSJTF) 1931 Jefferson Davis Highway Crystal Mall 3, Suite 104 Arlington, VA22202	10. SPONSOR/MONITOR'S ACRONYM(S) 11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
APUBLIC RELEASE
,

13. SUPPLEMENTARY NOTES

14. ABSTRACT
See Report.

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19. NAME OF RESPONSIBLE PERSON
			Public Release	59	http://www.acq.osd.mil/osjtf/library/library_alpha.html (blank) lfenster@dtic.mil
a. REPORT Unclassified	b. ABSTRACT Unclassified	c. THIS PAGE Unclassified			19b. TELEPHONE NUMBER International Area Code Area Code Telephone Number 703767-9007 DSN 427-9007

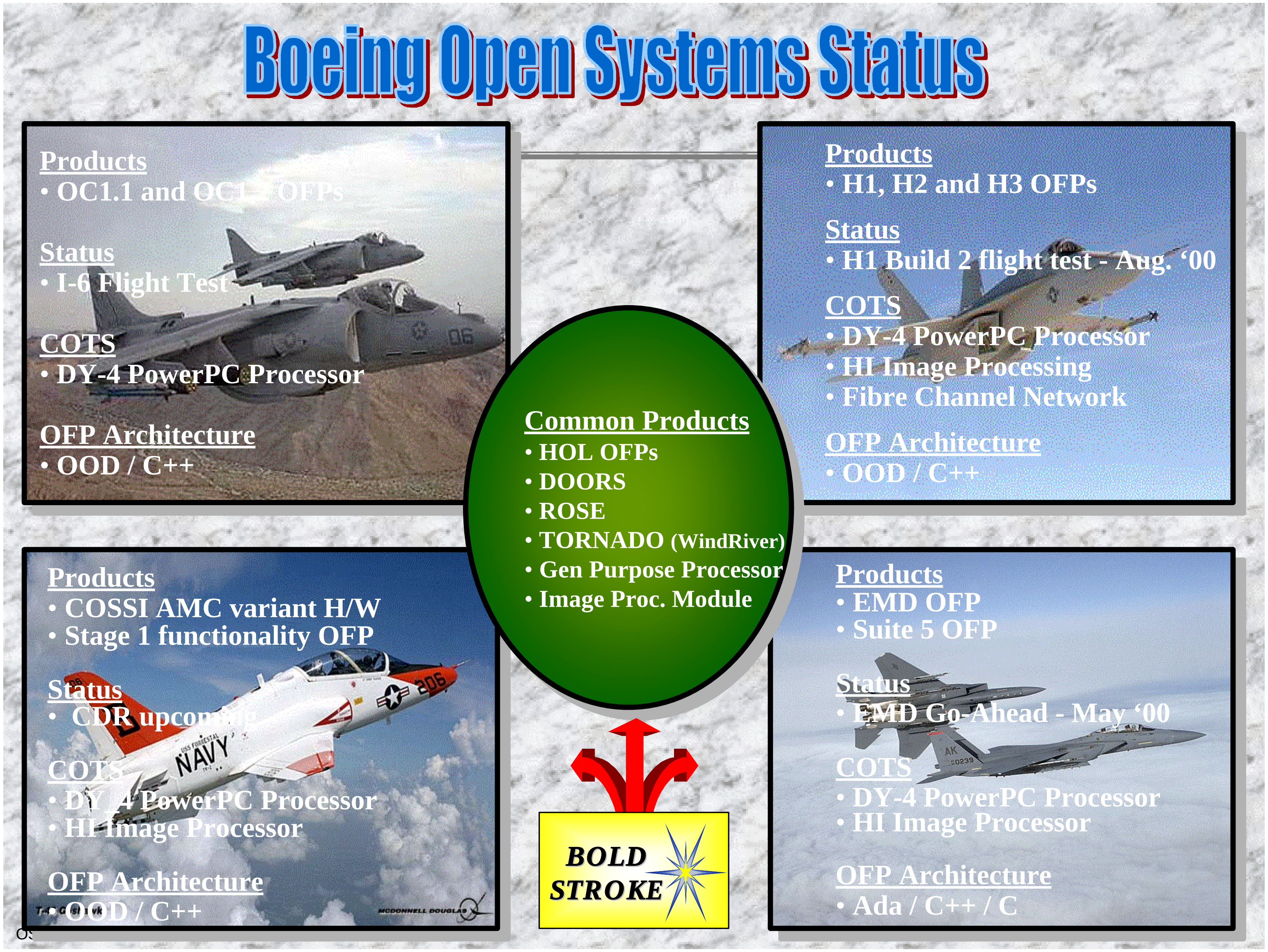
Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std Z39.18

Lessons Learned Agenda

0900-0915	Welcome (D. Weissgerber/J. Wojciehowski)
0915-1045	OSCAR Program (D. Weissgerber)
	Early Expectations & Assumptions
	Actual Experiences
1045-1100	Break
1100-1130	OSCAR Hardware (B. Abendroth)
1130-1145	Tools (C. Hibler)
1145-1200	Summary (D. Weissgerber)
1200-1300	Lunch

Lessons Learned Agenda

1300-1400	***Bold Stroke***
	OASIS (D. Seal)
	Cost Performance & Metrics (E. Beckles)
1400-1500	***Open Discussions***
1500	***Closing Remarks (D. Weissgerber/J. Wojciehowski)***


Boeing Open Systems Status
Products
• OC1.1 and OC1.2 OFPs
Status
• I-6 Flight Test
COTS
• DY-4 PowerPC Processor
OFP Architecture
• OOD / C++
Products
• H1, H2 and H3 OFPs
Status
• H1 Build 2 flight test - Aug. '00
COTS
• DY-4 PowerPC Processor
• HI Image Processing
• Fibre Channel Network
OFP Architecture
• OOD / C++
Common Products
• HOL OFPs
• DOORS
• ROSE
• TORNADO (WindRiver)
• Gen Purpose Processor
• Image Proc. Module
Products
• COSSI AMC variant H/W
• Stage 1 functionality OFP
Status
• CDR upcoming
COTS
• DY-4 PowerPC Processor
• HI Image Processor
OFP Architecture
• OOD / C++
Products
• EMD OFP
• Suite 5 OFP
Status
• EMD Go-Ahead - May '00
COTS
• DY-4 PowerPC Processor
• HI Image Processor
OFP Architecture
• Ada / C++ / C
BOLD STROKE

Boeing's Previous System Arch Lesson Learned Case Studies

- ***Software Modification/Maintenance Costs Are a Significant Recurring Investment***
- ***Must Break the Block Upgrade Paradigm Made Necessary by the Tight Coupling Between OFPs and Specific H/W Configurations***
- ***Assembly Language OFPs Have Become Increasingly Unstructured Through Many Upgrade Iterations***

OSCAR IPT Open System Lesson Learned Analysis

- ***Represents a Snapshot-In-Time***
  - ***Where We've Been***
  - ***Where We Are***
  - ***Where We're Going***
- ***Compiled by the Engineers Working the Issues***
  - ***Analysis of Key Impact Areas***
- ***Identifies Current Top 10 OSCAR Lessons Learned***
- ***Provides a Basis for Future Lessons Learned Comparisons/Analysis***

AV-8B OSCAR Principles

- ***Follow US DoD Directive For Acquisition Reform***
  - ***Apply Revised DoD Directive 5000 (dated 15 Mar 96)***
  - ***Commercial Business Philosophy***
  - ***Performance Based Specs vs Procurement Specs***
- ***Insert Commercial Technologies***
  - ***COTS Hardware***
  - ***COTS Software Development Environment***
- ***Reduce Life Cycle Cost***
- ***Apply Open System Architecture***
  - ***Emphasis on Non-Proprietary Hardware and Software***
  - ***Object Oriented Design and High Order Language***
  - ***Software Independent of Hardware***
- ***Increase Allied Software Development Workshare***

Review of Early Expectations

- ***OSCAR's Goals***
  - ***Reduce Life Cycle Support Cost of Software Upgrades (Cost Savings to be Realized during 3rd Block Upgrade)***
    - *Shortened OFP Development Cycle*
    - *Reduce Rework in Dev Cycle & DT/OT*
    - *Reduce Regression Testing in OC1.2 (OC1.1 set baseline)*
  - ***Leverage Commercial Technology***
  - ***Incorporate an Open Architecture Concept***
  - ***No Reduction in System Performance***

Review of OSCAR Open System Assumptions

- ***Implementation of Open Systems H/W and S/W Requires Up-Front Investment***
  - ***Recoupment Within** **<u>2-3 Updates</u>** **to the S/W***
- ***Open System Computing H/W is Based on Commercial Standards***
  - ***Promotes Competition***
  - ***Takes Advantage of Commercially Driven Requirements for Technology Insertion***
- ***LCC Analysis Shows a 30-40% Cost Reduction in Core Computing H/W and S/W Development but not necessarily applicable to System Integration/Test of Multi-Sys Block Upgrades***

Review of OSCAR Open System Assumptions (cont.)

- ***OSCAR and Open Systems Computing Does Not Affect Tasks Associated with the Airframe or Flight Qualification of New Weapons/Capabilities***
- ***Two-Level Maintenance Concept Philosophy Will Reduce LCC and Increase Operational Availability***
- ***OSA provides Arch for a Plug-and-Play Trainer Concept***
- ***With OSCAR as First Large Scale Implementation of Open Systems and Object Oriented S/W:***
  - ***Reluctance to Fully Realize the Cost Benefits Until OSCAR is Fielded and all the Data Collected and Analyzed***

Review of OSCAR's Open System Assumptions (cont.)

- ***OSCAR's Open System Architecture Will Make Incremental Upgrades Possible by Decoupling H/W and S/W (I.e., MSC-750-G4)***
- ***Commercial Off-The-Shelf Products can be Directly Incorporated with Minimal Development Costs***
  - ***Multi-Vendor Support Ensures Competitive Procurement Costs***
- ***Software LCC Savings are Derived from the High Degree of Modularity Envisioned***
  - ***Less Than Half the Regression Test and Re-Qual Effort of Today***

Data & Metrics Currently Collected

- ***SPI***
- ***CPI***
- ***Requirements -- System & software levels, stability index***
- ***SLOC -- Estimates vs. actuals, productivity factor***
- ***Classes***
- ***Peer Review***
- ***TWD -- Development & ground test execution***
- ***Flight Test -- flights, test points, analysis***
- ***Problem Reports - various flavors***
- ***Throughput & Memory Spare***
- ***Hardware Performance***
- ***Risk***

Initial Expectations for Metrics

- ***SPI -- Identify an immediate schedule problem***
- ***CPI -- Control overspending, identify underruns***
- ***System & Software Requirements -- Track the development to plan and identify any Growth***
- ***Requirements Stability -- Control requirements growth***
- ***SLOC Actuals vs. Estimated -- Control growth and 'gold-plating'***
- ***Software productivity (Manhrs/SLOC) -- Improve efficiency within which software is produced***
- ***Classes Actuals vs. Planned To Date -- Indication of performance to schedule***
- ***Peer Review -- Capture errors before the product is delivered***

Initial Expectations of Metrics

- ***TWD Development & Ground Test -- Readiness of test team to support system level test phase***
- ***Problem Reports -- Quality of the software & where are problems found***
- ***Throughput/Memory -- Keep software within the bounds of hardware performance***
- ***Risk -- Control risks & be prepared to act quickly if they materialize***

What Metrics Actually Provided

- **SPI -- Watch The Details**
  - *Lower level problems are masked within larger cost accounts*
  - *Top-level SPI can mask lower level account SPI difficulties*
  - *Provides good focus for the CAMs*

Overall Program Healthy

Critical Path Behind Schedule

OSCAR OC1.1 PERFORMANCE STOP LIGHT CHART

EEK ENDING: 27FEB00													
							⇧		⇧				
OTAL CONVERSION (TDL01)							⇧		⇧				
AVIONICS SOFTWARE													
o WEAPON DELIVERY							⇧		⇧				
o PILOT/VEHICLE INTERFA							⇧		⇩				
o SENSORS & TARGET							⇩		⇩				
o NAVIGATION & A/C STAT	DMU	REBHAN	(15)	[illegible]	9	414	⇩	100.1	⇧	102.8	14,721	20,292	23,558
o STF/DTE	DMT	BELL, R.T.	[illegible]	7	523	704	⇧	107.6		110.5	6,727	9,987	9,988
o SSEE	DMR	RUSSELL, W.H	0	(15)	0	(428)		100.0	⇩	91.0	4,766	9,166	10,866
o QUALITY	DMR	HIBLER, C.A	0	3	0	26		100.0	⇧	102.5	1,048	2,019	2,135
o SOFTWARE BULK	DMR	HIBLER, C.A.	0	0	0	2,969		100.0		137.9	7,836	10,805	8,668
TACTICAL ELECT WARFARE	DCA	MARX, J.A.	0	5	0	82		100.0	⇧	103.6	2,251	3,626	3,626
AVIONICS HARDWARE													
o MSC/CD	DEA	ABENDROTH	0	5	(10)	186		99.6	⇧	107.8	2,376	3,527	3,616
o WMC	DEA	SZCZUKA	0	(4)	(98)	425		97.9	⇩	110.4	4,080	6,396	6,397
INTEGRATION													
o EMC	DHF	GOODWIN	0	12	0	161		100.0	⇧	122.5	714	1,303	1,305
AVIONICS TEST	J00	ILGES, J.F.	70	173	(188)	1,285	⇧	98.8	⇧	109.4	13,655	21,701	21,367
LOGISTICS (Product Supt)	SK0	HERBERT, B.K.	0	7	0	63		100.0	⇧	107.8	800	2,038	2,038
DWG RELEASE	AB0	REARDON	0	1	0	673		100.0	⇩	177.5	868	2,031	1,356
SYSTEM ENGINEERING	BD0	WESTPHAL, J.L.	0	7	0	825		100.0	⇧	209.8	752	2,097	1,486
SAFETY/R&M	BD0	MCCOY, R.L.	0	(9)	0	484		100.0	⇩	136.5	1,325	3,211	2,630
MANAGEMENT	DMR	FRANKENFIELD, C.R	0	(40)	0	(387)		100.0	⇩	94.4	6,877	11,648	12,133
GENERAL BULK	DMR	RUSSO, A.G.	0	36	0	(448)		100.0	⇧	93.7	7,100	11,154	11,154
OTAL AMRAAM (TDL02)	H00		86	44	47	1,251	⇧	100.1	⇧	103.2	39,432	62,602	65,343
AVIONICS SOFTWARE													
o WEAPON DELIVERY	DMW	HEZEL, K.C.	15	18	43	866	⇧	100.6	⇧	114.4	6,005	14,168	14,993
o PILOT/VEHICLE INTERFA	DMY	VOLLE, D.A.	0	44	0	(1,241)		100.0	⇧	86.5	9,213	9,271	9,775
o SENSORS & TARGET	DMX	SHYLANSKI, J.	0	22	0	(266)		100.0	⇧	97.3	9,891	15,150	17,669
o STF/DTE	DMT	BELL, R.T.	0	6	0	169		100.0	⇧	146.2	365	573	573
o SOFTWARE BULK	DMR	HIBLER, C.A.	0	0	0	545		100.0		153.9	1,010	1,555	1,093
INTEGRATION													
o EMC	DHF	GOODWIN	0	3	0	111		100.0	⇧	156.6	196	361	298
AVIONICS TEST	J00	ILGES, J.F.	71	(53)	4	940	⇧	100.1	⇩	114.0	6,696	11,294	10,896
LOGISTICS (Product Supt)	SK0	HERBERT, B.K.	0	3	0	30		100.0	⇧	111.2	265	567	609
AIR VEHICLE TECHNOLOGY	BAJ	BRIGULIO, J.A.	0	(12)	0	(91)		100.0	⇩	94.8	1,754	1,807	2,010
DWG RELEASE	AB0	REARDON	0	0	0	155		100.0	⇩	168.5	226	497	386
SYSTEM ENGINEERING	BD0	WESTPHAL, J.L.	0	1	0	288		100.0	⇩	152.1	552	1,422	1,032
MANAGEMENT	DMR	FRANKENFIELD, C.R	0	15	0	54		100.0	⇧	104.5	1,199	3,104	2,941
GENERAL BULK	DMR	RUSSO, A.G.	0	(2)	0	(307)		100.0		85.1	2,060	2,833	3,068
OTAL 1760B (TDL03)	H00		(6)	27	(14)	(15)		99.9	⇧	99.9	14,868	22,121	22,050
AVIONICS SOFTWARE													
o WEAPON DELIVERY	DMW	HEZEL, K.C.	3	45	154	507	⇩	107.2	⇧	128.3	1,790	5,907	5,019
o SOFTWARE BULK	DMR	HIBLER, C.A.	0	0	0	71		100.0		999.9	1	72	42
INTEGRATION													
o EMC	DHF	GOODWIN	0	6	0	33		100.0	⇧	122.8	143	448	449
AVIONICS TEST	J00	ILGES, J.F.	1	38	(17)	539	⇧	99.6	⇧	115.7	3,435	4,657	4,521
LOGISTICS (Product Supt)	SK0	HERBERT, B.K.	0	(9)	0	(212)		100.0	⇩	91.3	2,425	3,023	3,260
DWG RELEASE	AB0	REARDON	0	(1)	0	(221)		100.0		16.9	266	64	213
AIRFRAME/SUBSYSTEM	ACK	HOHL, K.L.	(11)	(50)	(151)	(622)	⇩	96.8	⇩	88.2	5,259	5,432	5,520

What Metrics Actually Provided

- ***CPI -- New functionality Costs More Than Legacy***

New Functionality

OSCAR OC1.1 PERFORMANCE STOP LIGHT CHART

EEK ENDING: 27FEB00	TEAM	TEAM LEADER	SVdelta	CVdelta	SV	CV		SPI		CPI	ACI	BAC	EAC
GRAND TOTAL			348	554	(1,858)	12,301	⇧	99.2	⇧				
OTAL CONVERSION (TDL01)	H00						⇧		⇧				
AVIONICS SOFTWARE													
o WEAPON DELIVERY							⇧		⇧				
o PILOT/VEHICLE INTERFA									⇩				
o SENSORS & TARGET							⇩		⇩				
o NAVIGATION & A/C STAT	DMU	REBHAN	(15)	81	9	414	⇩	100.1	⇧	102.8	14,721	20,292	23,558
o STF/DTE	DMT	BELL, R.T.	30	7	523	704	⇧	107.6		110.5	6,727	9,987	9,988
o SSEE	DMR	RUSSELL, W.H	0	(15)	0	(428)		100.0	⇩	91.0	4,766	9,166	10,866
o QUALITY	DMR	HIBLER, C.A.	0	3	0	26		100.0	⇧	102.5	1,048	2,019	2,135
o SOFTWARE BULK	DMR	HIBLER, C.A.	0	0	0	2,969		100.0		137.9	7,836	10,805	8,668
TACTICAL ELECT WARFARE	DCA	MARX, J.A.	0	5	0	82		100.0	⇧	103.6	2,251	3,626	3,626
AVIONICS HARDWARE													
o MSC/CD	DEA	ABENDROTH	0	5	(10)	186		99.6	⇧	107.8	2,376	3,527	3,616
o WMC	DEA	SZCZUKA	0	(4)	(98)	425		97.9	⇩	110.4	4,080	6,396	6,397
INTEGRATION													
o EMC	DHF	GOODWIN	0	12	0	161		100.0	⇧	122.5	714	1,303	1,305
AVIONICS TEST	J00	ILGES, J.F.	70	173	(188)	1,285	⇧	98.8	⇧	109.4	13,655	21,701	21,367
LOGISTICS (Product Supt)	SK0	HERBERT, B.K.	0	7	0	63		100.0	⇧	107.8	800	2,038	2,038
DWG RELEASE	AB0	REARDON	0	1	0	673		100.0	⇩	177.5	868	2,031	1,356
SYSTEM ENGINEERING	BD0	WESTPHAL, J.L.	0	7	0	825		100.0	⇧	209.8	752	2,097	1,486
SAFETY/R&M	BD0	MCCOY, R.L.	0	(9)	0	484		100.0	⇩	136.5	1,325	3,211	2,630
MANAGEMENT	DMR	FRANKENFIELD, C.R	0	(40)	0	(387)		100.0	⇩	94.4	6,877	11,648	12,133
GENERAL BULK	DMR	RUSSO, A.G.	0	36	0	(448)		100.0	⇧	93.7	7,100	11,154	11,154
OTAL AMRAAM (TDL02)	H00		86	44	47	1,251	⇧	100.1	⇧	103.2	39,432	62,602	65,343
AVIONICS SOFTWARE													
o WEAPON DELIVERY	DMW	HEZEL, K.C.	15	18	43	866	⇧	[illegible]	⇧	114.4	6,005	14,168	14,993
o PILOT/VEHICLE INTERFA	DMY	VOLLE, D.A.	0	44	0	(1,241)		[illegible]	⇧	86.5	9,213	9,271	9,775
o SENSORS & TARGET	DMX	SHYLANSKI, J.	0	22	0	(266)		100.0	⇧	97.3	9,891	15,150	17,669
o STF/DTE	DMT	BELL, R.T.	0	6	0	169		100.0	⇧	146.2	365	573	573
o SOFTWARE BULK	DMR	HIBLER, C.A.	0	0	0	545		100.0		153.9	1,010	1,555	1,093
INTEGRATION													
o EMC	DHF	GOODWIN	0	3	0	111		100.0	⇧	156.6	196	361	298
AVIONICS TEST	J00	ILGES, J.F.	71	(53)	4	940	⇧	100.1	⇩	114.0	6,696	11,294	10,896
LOGISTICS (Product Supt)	SK0	HERBERT, B.K.	0	3	0	30		100.0	⇧	111.2	265	567	609
AIR VEHICLE TECHNOLOGY	BAJ	BRIGULIO, J.A.	0	(12)	0	(91)		100.0	⇩	94.8	1,754	1,807	2,010
DWG RELEASE	AB0	REARDON	0	0	0	155		100.0	⇩	168.5	226	497	386
SYSTEM ENGINEERING	BD0	WESTPHAL, J.L.	0	1	0	288		100.0	⇩	152.1	552	1,422	1,032
MANAGEMENT	DMR	FRANKENFIELD, C.R	0	15	0	54		100.0	⇧	104.5	1,199	3,104	2,941
GENERAL BULK	DMR	RUSSO, A.G.	0	(2)	0	(307)		100.0		85.1	2,060	2,833	3,068
OTAL 1760B (TDL03)	H00		(6)	27	(14)	(15)		99.9	⇧	99.9	14,868	22,121	22,050
AVIONICS SOFTWARE													
o WEAPON DELIVERY	DMW	HEZEL, K.C.	3	45	154	507	⇩	107.2	⇧	128.3	1,790	5,907	5,019
o SOFTWARE BULK	DMR	HIBLER, C.A.	0	0	0	71		100.0		999.9	1	72	42
INTEGRATION													
o EMC	DHF	GOODWIN	0	6	0	33		100.0	⇧	122.8	143	448	449
AVIONICS TEST	J00	ILGES, J.F.	1	38	(17)	539	⇧	99.6	⇧	115.7	3,435	4,657	4,521
LOGISTICS (Product Supt)	SK0	HERBERT, B.K.	0	(9)	0	(212)		100.0	⇩	91.3	2,425	3,023	3,260
DWG RELEASE	AB0	REARDON	0	(1)	0	(221)		100.0		16.9	266	64	213
AIRFRAME/SUBSYSTEM	ACK	HOHL, K.L.	(11)	(50)	(151)	(622)	⇩	96.8	⇩	88.2	5,259	5,432	5,520

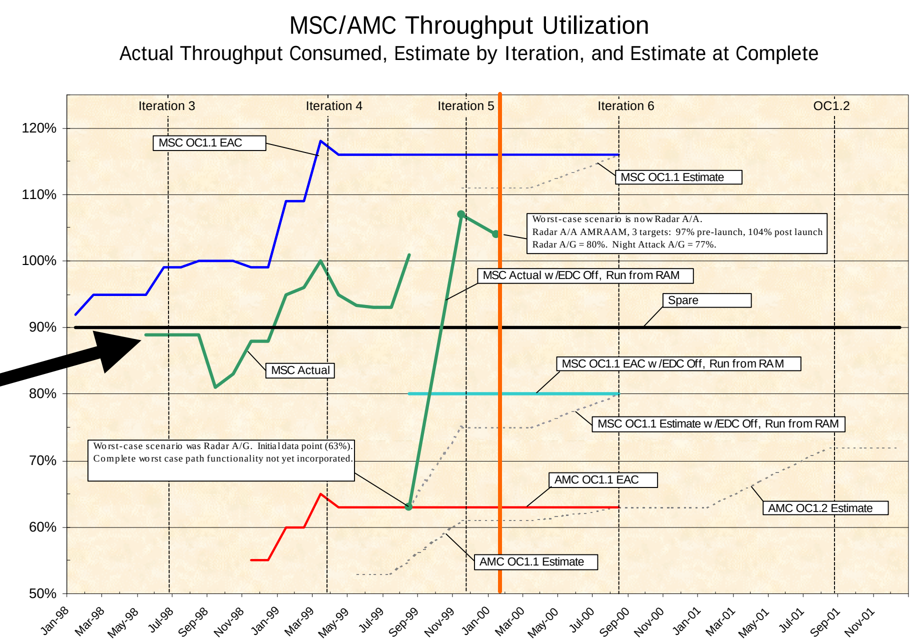
What Metrics Actually Provided

- ***System Requirements - No Changes Resulting From OO/C++ Development***
  - ***Level Of Detail & Complexity Commensurate With Assembly***
  - ***OO Makes Traceablity To Code Is Difficult (see other chart)***
- ***Requirements Stability -- good to show what's moving through the system, but don't really know how many requirements and corresponding code/tests are affected (traceability)***
- ***Risks -- hard to maintain a monthly review juggling schedules, but good tool to keep on top of issues, when High risks are identified - resources are focused on them***
  - *Engineers tend to set risks at HW/SW detail level and not see the top level System Functionality High Risks*

What Metrics Actually Provided

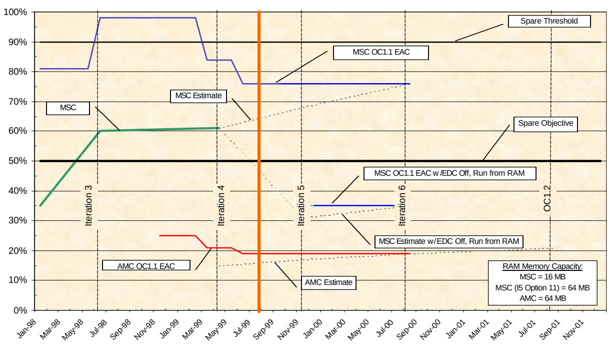
- ***Throughput Usage***
  - ***OO , COTS OS makes throughput consumption difficult to predict***

Predicted Usage

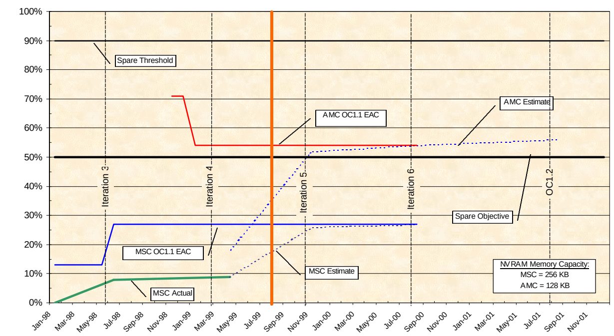


What Metrics Actually Provided

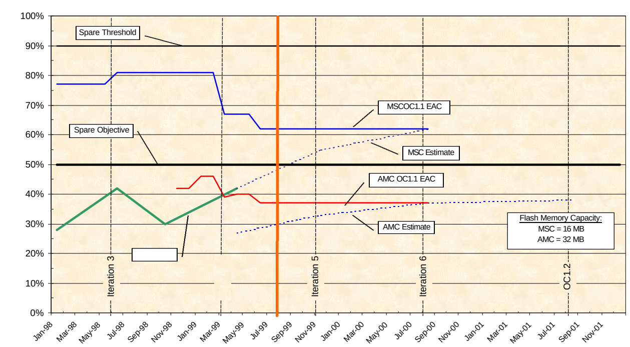
MSC/AMC RAM Memory Utilization
Actual Memory Consumed, Estimate by Iteration, and Estimate at Complete



MSC/AMC NVRAM Memory Utilization
Actual Memory Consumed, Estimate by Iteration, and Estimate at Complete



MSC/AMC Flash Memory Utilization
Actual Memory Consumed, Estimate by Iteration, and Estimate at Complete

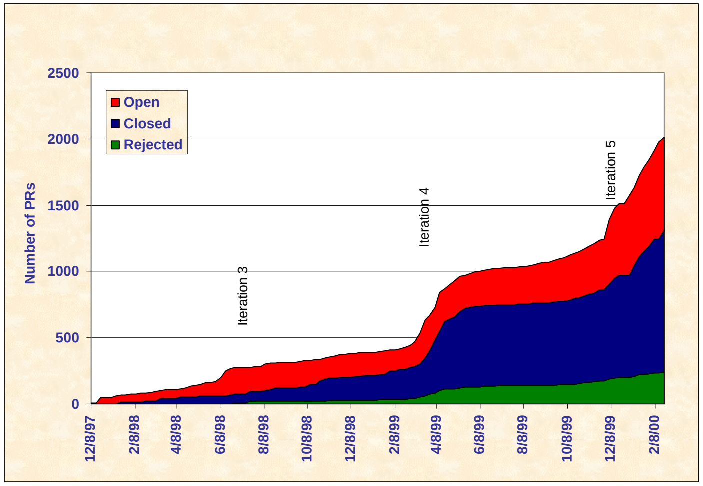


- ***Memory Usage***
  - ***Consumption can be predictably scaled from assembly language implementation***

What Metrics Actually Provided

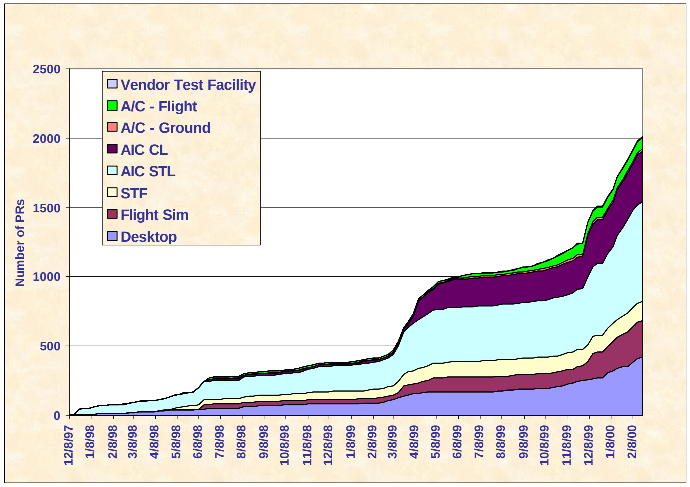
Problem Reports -- Open/Closed/Rejected

– *OO/C++ enables <u>trained</u> developers with Tools to rapidly diagnose and correct anomalies.*



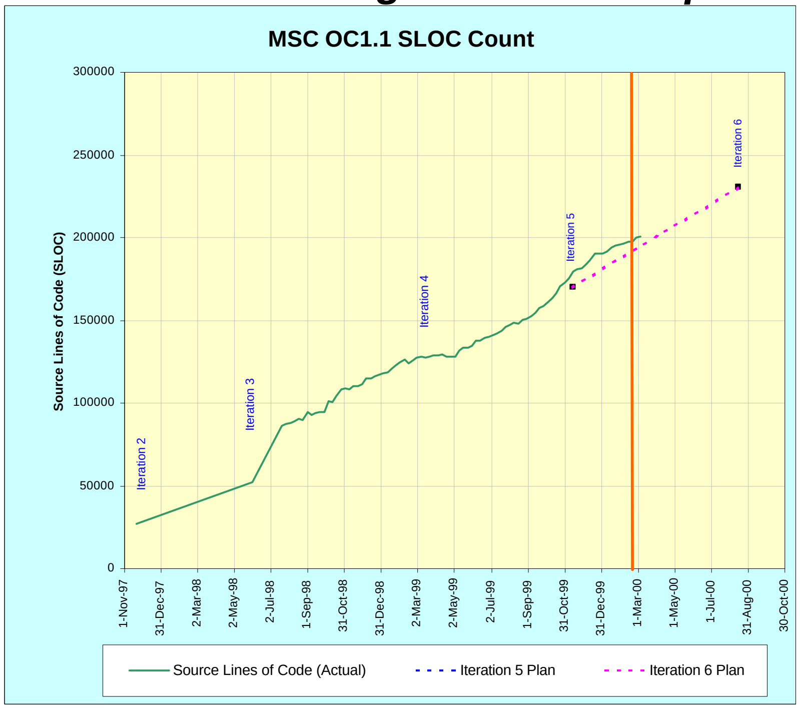
What Metrics Actually Provided

- **Problem Reports - Where Found**
  - DTE Saves Time & Money
  - Provides a "Software Test Facility" on every desktop
  - Less problems found in flight than Legacy OFP



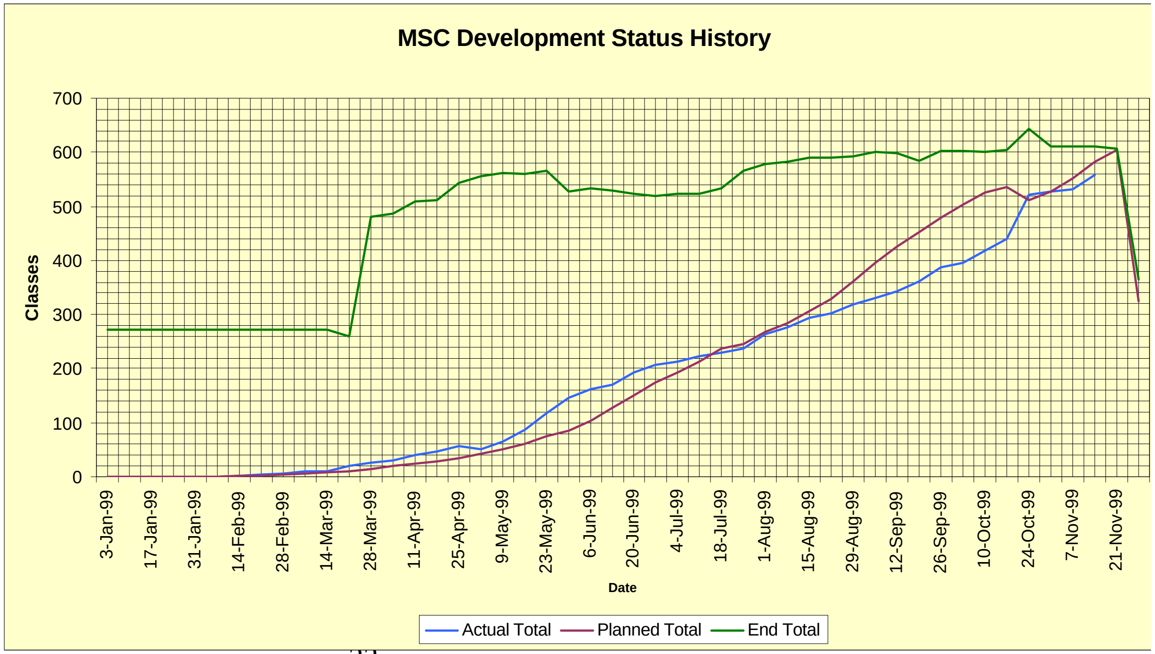
What Metrics Actually Provided

- **SLOC**
  - *Not very useful*
    - *Some code "auto"-generated by 4th generation tools*
  - *Poor unit for estimating resources required*



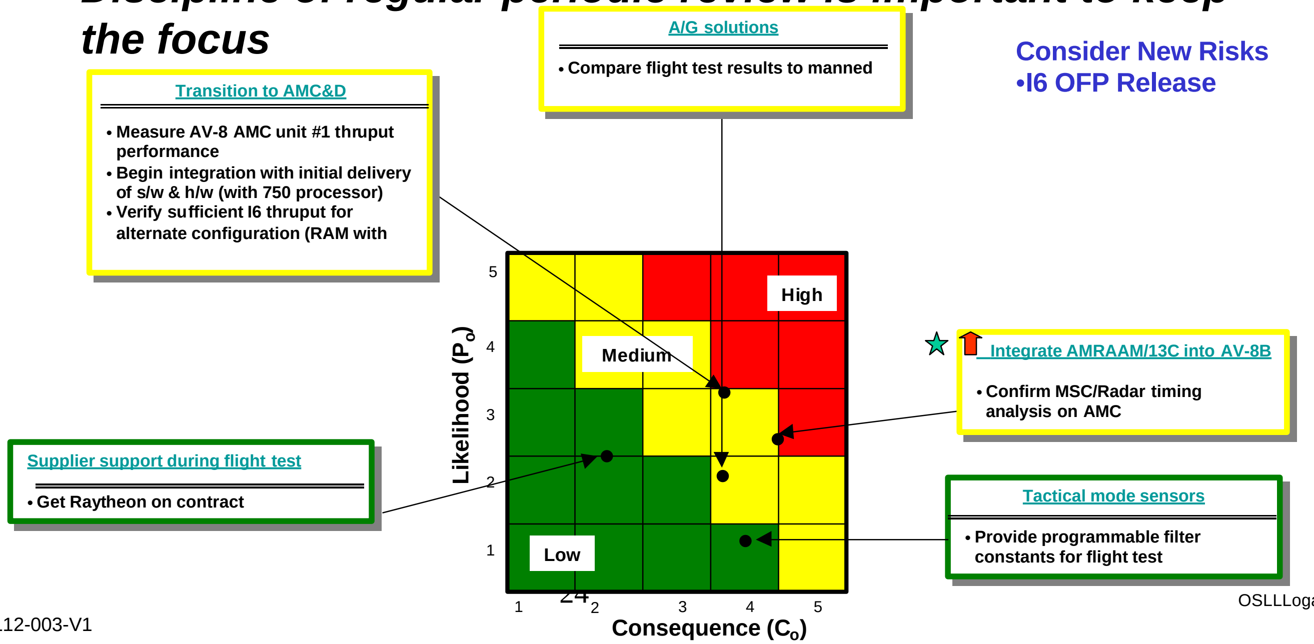
What Metrics Actually Provided

- ***Classes***
  - ***Best measure of development progress***
    - *Similar to function points*
    - *SLOC difficult to estimate*



What Metrics Actually Provided

- ***Risk***
  - *Good tool to keep on top of issues but can bring too much Political help*
    - *When high risks are identified -- resources are focused on them*
  - *Discipline of regular periodic review is important to keep the focus*



Summary of OS Lessons Learned For Currently Collected Metrics

- ***SPI -- Watch The Details***
- ***CPI -- New functionality Costs More Than Legacy***
- ***System Requirements - No Changes For Assembly***
  - ***Traceablity To Code Is Difficult***
- ***TWD Development -- Same as in Traditional Development***
- ***SLOC count -- Not as Useful for OO/C++ Development Tracking***
- ***Classes -- Good Indicator of Development Progress***

Summary of OS Lessons Learned For Currently Collected Metrics

- ***Problem Reports - Total -- OO/C++ a Benefit to Problem Resolution***
- ***Problem Reports - Where found -- DTE Saves Time & Money***
- ***Throughput Usage - OO, COTS Makes Prediction Difficult***
- ***Memory Usage - Scaleable from Legacy Development***
- ***Risk - Good Tool to Focus Attention & Resources, if Risk Identification doesn't get too Political***

Technology Challenges

COTS supports the code/debug/unit test stages of development well but many <u>Voids</u> still exist:

- ***"Front end" of process***
  - ***Model-based tools for requirements/design capture***
  - ***Automated configuration and integration of components***
- ***"Back end" of process***
  - ***Simulation-based testing***
- ***Support for hard real-time embedded systems is limited***
  - ***Quality-of-service requirements expression/guarantees***
- ***Legacy system constraints***
  - ***Infusing new technology into resource-limited, "closed" systems***
- ***High Integrity System development technologies***

Cultural Challenges

- *Acquisition culture presents impediments as well*
  - *"Silo" approach to planning/funding system modernization*
  - *"Wasn't invented here" mindset in programs*
  - *Inability to trade front-end investment for life-cycle returns, even when business case is compelling*
  - *Synergy with COTS industry will always be limited without cultural transformation*
  - *Support structure based on single fielded configuration*
  - *T&E community resistance to tailored re-qualification*

No incentive for multi-platform development

OSA Lessons Learned - Standards

Goal: **Use Widely Accepted Commercial Standards**

- **Standardize Module Form, Fit, Function and Interface (F^3I) to Allow Functional Performance Upgrades**
- **USE COTS Standards for Networks, Processors, Memory, and Operating System**

Reality: **Existing Commercial Standards Do Not Typically Accommodate Aerospace Requirements**

- **Real Time Operation - Flight Dynamics**
- **Memory Partitioning for Fault Containment**
- **Built-In-Test**

Solution: **Modify Commercial Standards Through Active Participation in Standards Bodies**

- **ANSI Fibre Channel Avionics Environment (FC-AE)**
- **Modify Commercial STD Common Object Request Broker Architecture (CORBA) for Real-Time Operation**
- **Add Service Layers on Top of Commercial Software Infrastructure**

OSA Lessons Learned - Specifications

Goal: **Focus on Specifying Functional/Performance Requirements versus "How To"**

- **Use Commercial Specs Wherever Possible**
- **Use Tailored Mil-Specs**
- **Eliminate Unnecessary "How To" specs**

Reality: **It is Difficult to Prevent Engineers (Boeing, Customer, and Supplier) From Diving Down Into Too Much Detail**

- **Commercial Specifications may not match Aerospace requirements**
- **Additional effort needed to ensure Performance Levels and interoperability Are Achievable**

Solution: **Need to get a Better Handle on the High Level Performance Requirements**

- **Develop benchmark application program to validate memory and throughput for COTS processors**
- **Using a "Performance Prediction Team" to Conduct Simulation and Modeling of Key System Attributes.**
- **Evaluate Lab Prototype H/W to Gather Data.**

COTS Lessons Learned

- ***COTS May Not Work As Well For Your Application As The Application For Which It Was Developed***
- ***COTS Frequently Has Surprises, Especially With Little Used Features***
- ***COTS Documentation May Be Lacking, Or Will Not Tell You How It Will Work In Your System***

Lessons Learned - Diagnostics

- ***Diagnostics Processes/Tools must better address False Alarm Rate***
- ***Supplier must better understand Total Diagnostics Requirements***
  - ***Fault Coverage***
  - ***Fault Isolation***
  - ***False Alarms***
  - ***Failure Reporting & Recording***
- ***Diagnostic System must have integrated on-board and off-board capability that can be updated in a timely manner***

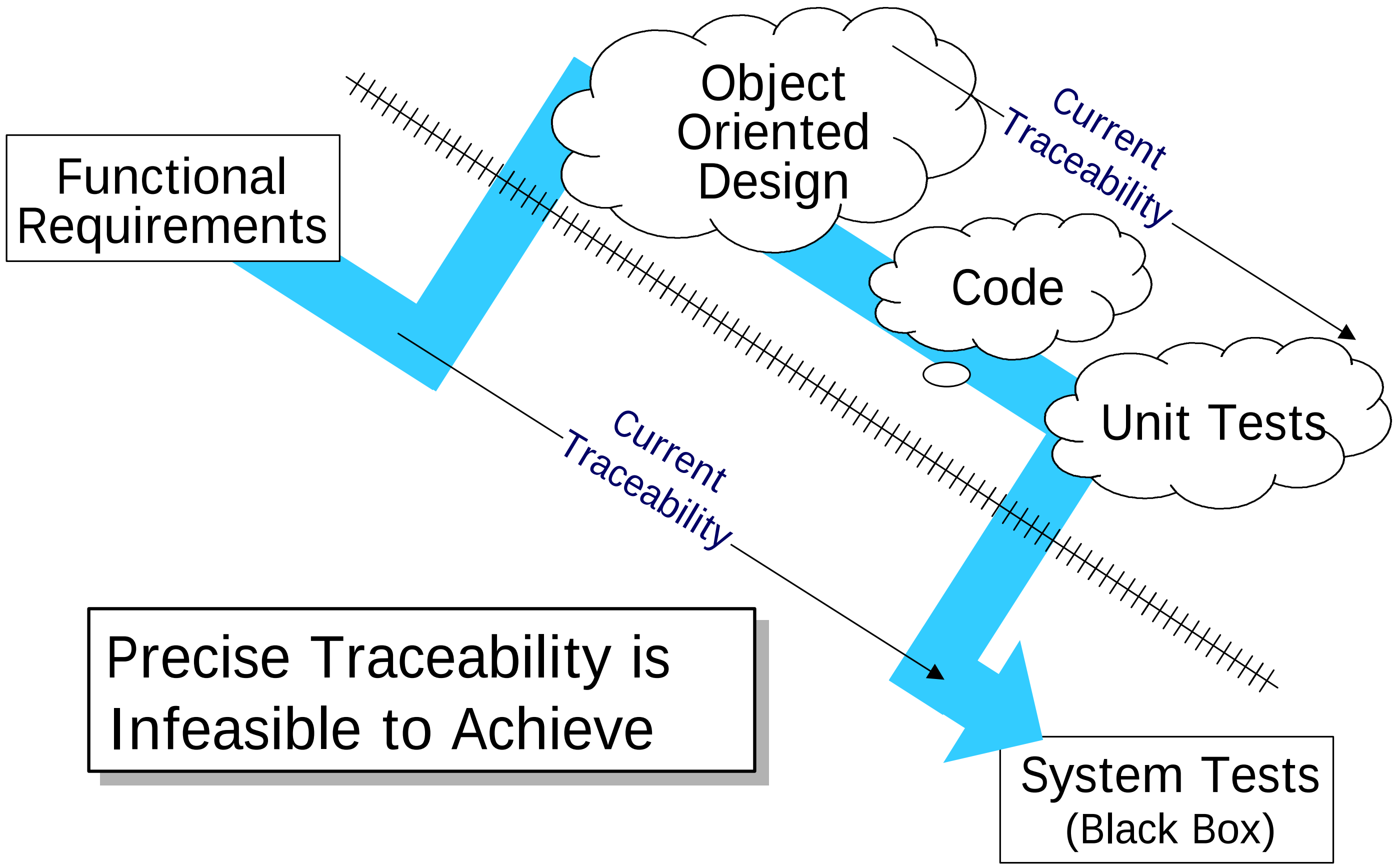
Total System Diagnostics Architecture Must Minimize NFF Occurrences

Lessons Learned - Prototyping

- **Early And Frequent Prototyping Required Throughout The Program**
- **Develop Software Incrementally Utilizing Daily Builds**
- **Complex Functionality needs to be partitioned and implemented early**
- **Verify Design And Ensure API's Meet Needs Of User**
- **Verify Software And Hardware Performing As Expected**

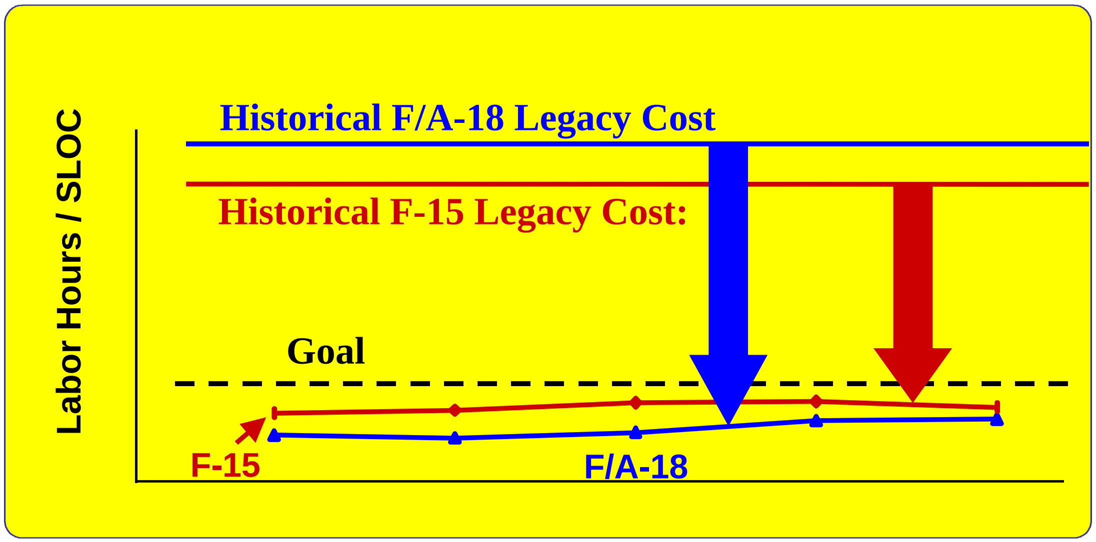
No New Lessons from Legacy Developments

Object Oriented Design in a Functional Decomposition World



Early Returns - Measured Benefit

Cumulative Software Development Productivity

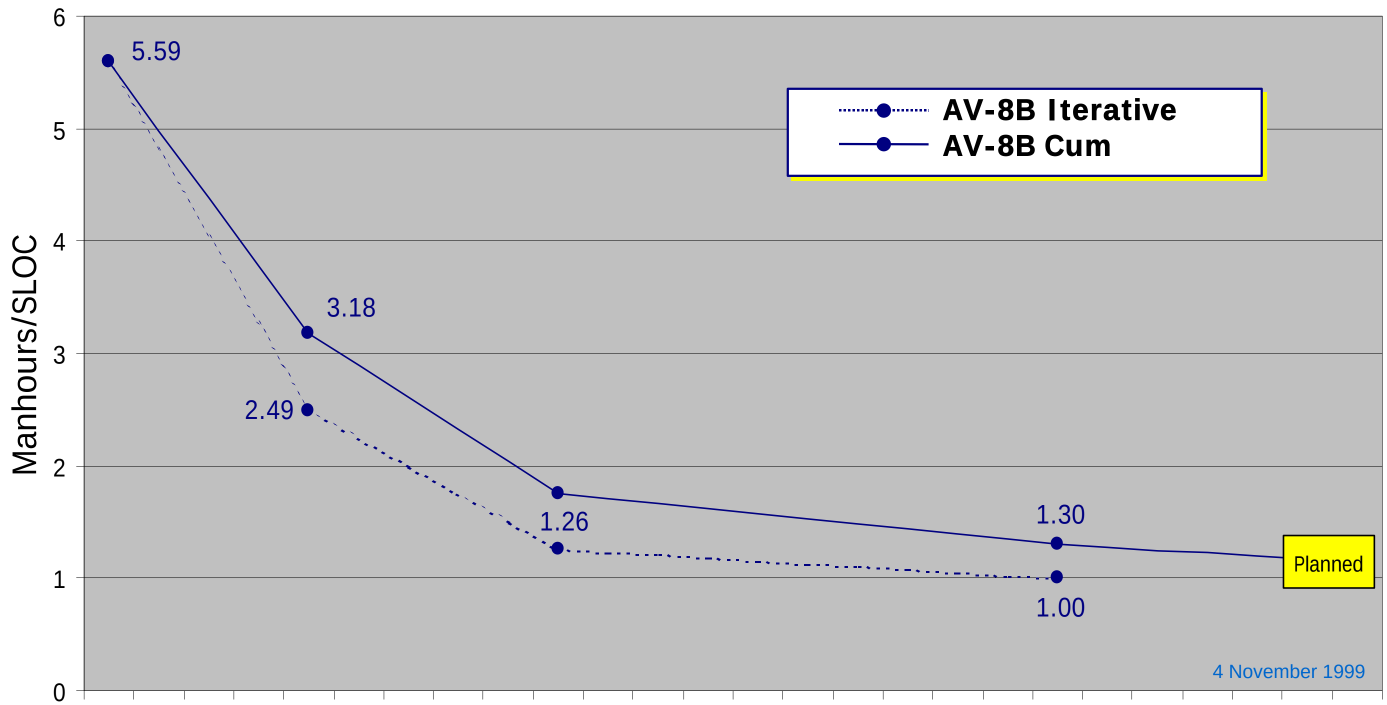


Key Sources of Gain:

- Reuse (of all types)
- COTS Tools
- Change Containment
- Desktop Testing
- High Order Language

Measured Software Development Affordability Improvement

S/W Development Productivity (Hand plus Rose Generated Code)



Lesson Learned - OSCAR Hardware

Qual Test

- ***The following environmental qual tests have been completed :***

MSC & WMC

- **Temp-Alt**
- **Vibration**
- **EMIC**
- **Acoustic Noise**
- **Loads**
- **Shock**
- **Humidity**
- **Salt**
- **Exp Atmosphere**
- **Sand & Dust**

Qual Test Cont'd

- ***COTS hardware did Well.***
  - ***No problems with off-the-shelf DY-4 Processor board (one capacitor failure in RDT.***
- ***No problems with plastic parts (PEMS)***
  - ***Hardware with plastic parts were exposed to MIL-STD-810 Humidity and Salt-Fog environments in two WRA's with no failures.***
  - ***Was a major concern of some people early in the program.***

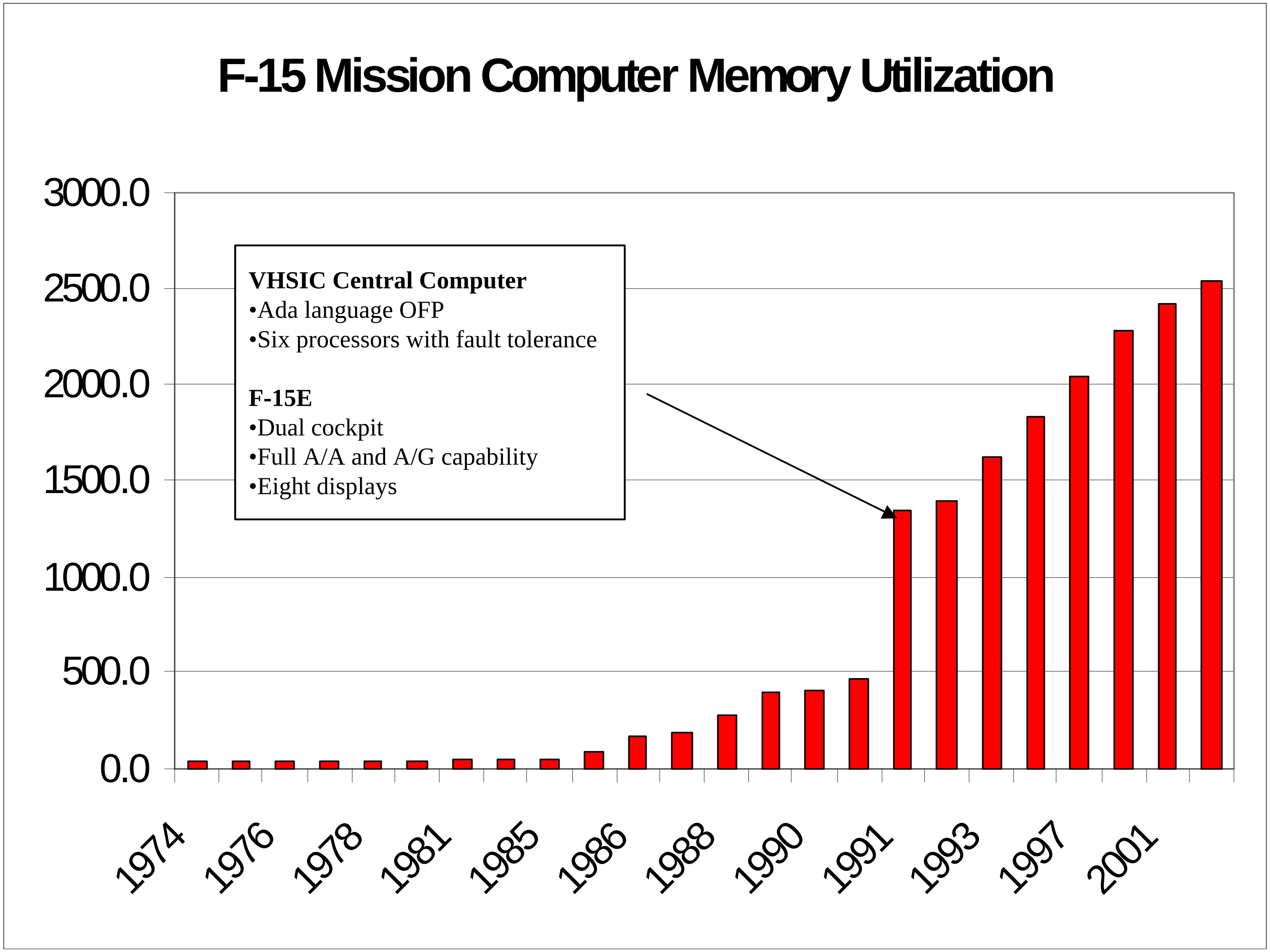
Reliability

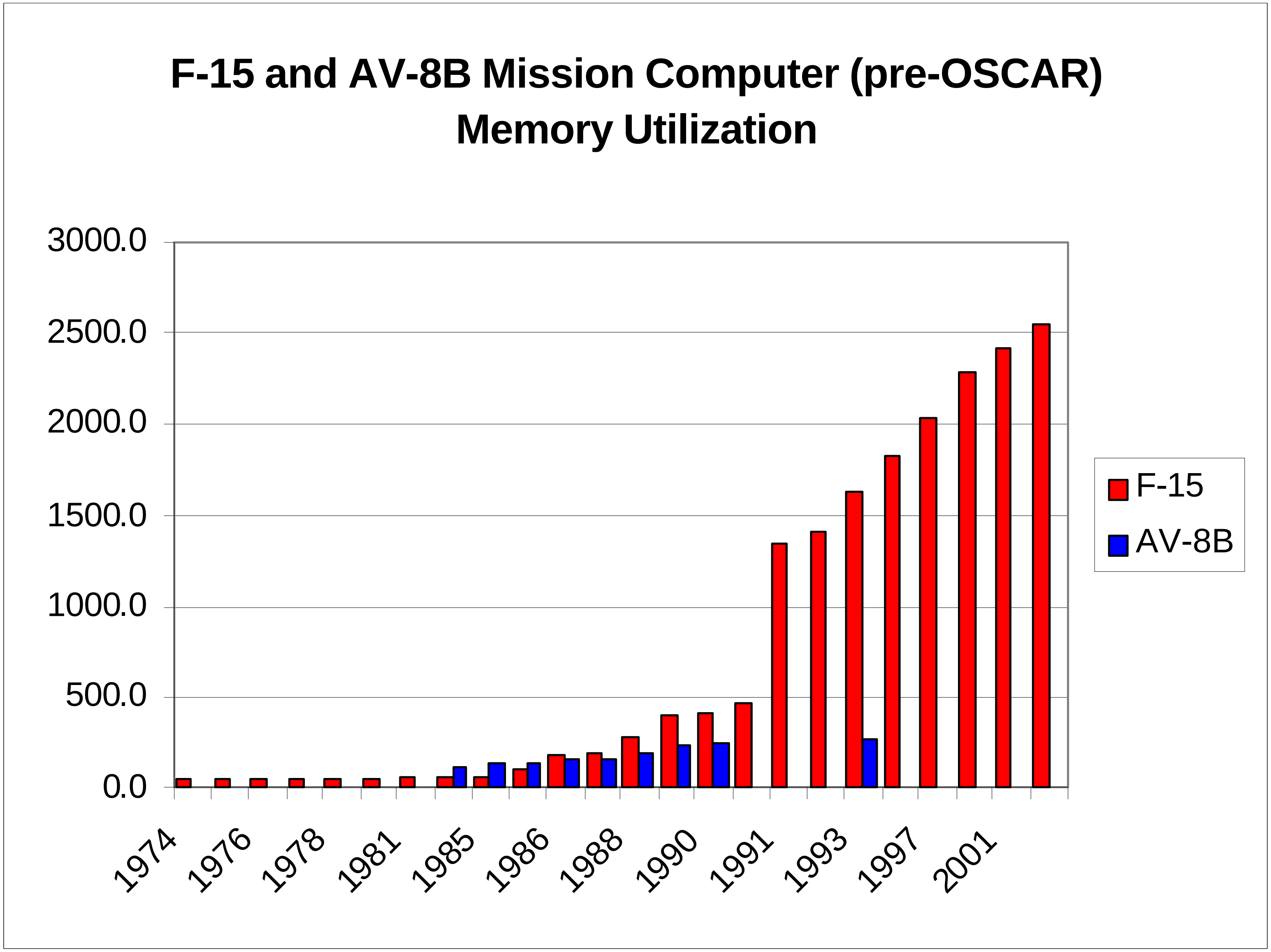
- ***Reliability experience to date with COTS hardware has been good.***
- ***Reliability Development Testing (RDT) done on three WRAs.***
  - ***WMC - 1,000+ hours***
  - ***MSC #1- 1,000+ hours***
  - ***MSC #2 - 1,000+ hours***
- ***One capacitor failure on COTS board, Root cause unknown.***
- ***One commercial grade capacitor failed on another SRA. Switching to a MIL-SPEC capacitor.***
- ***Other failures occurred, but unrelated to COTS hardware.***

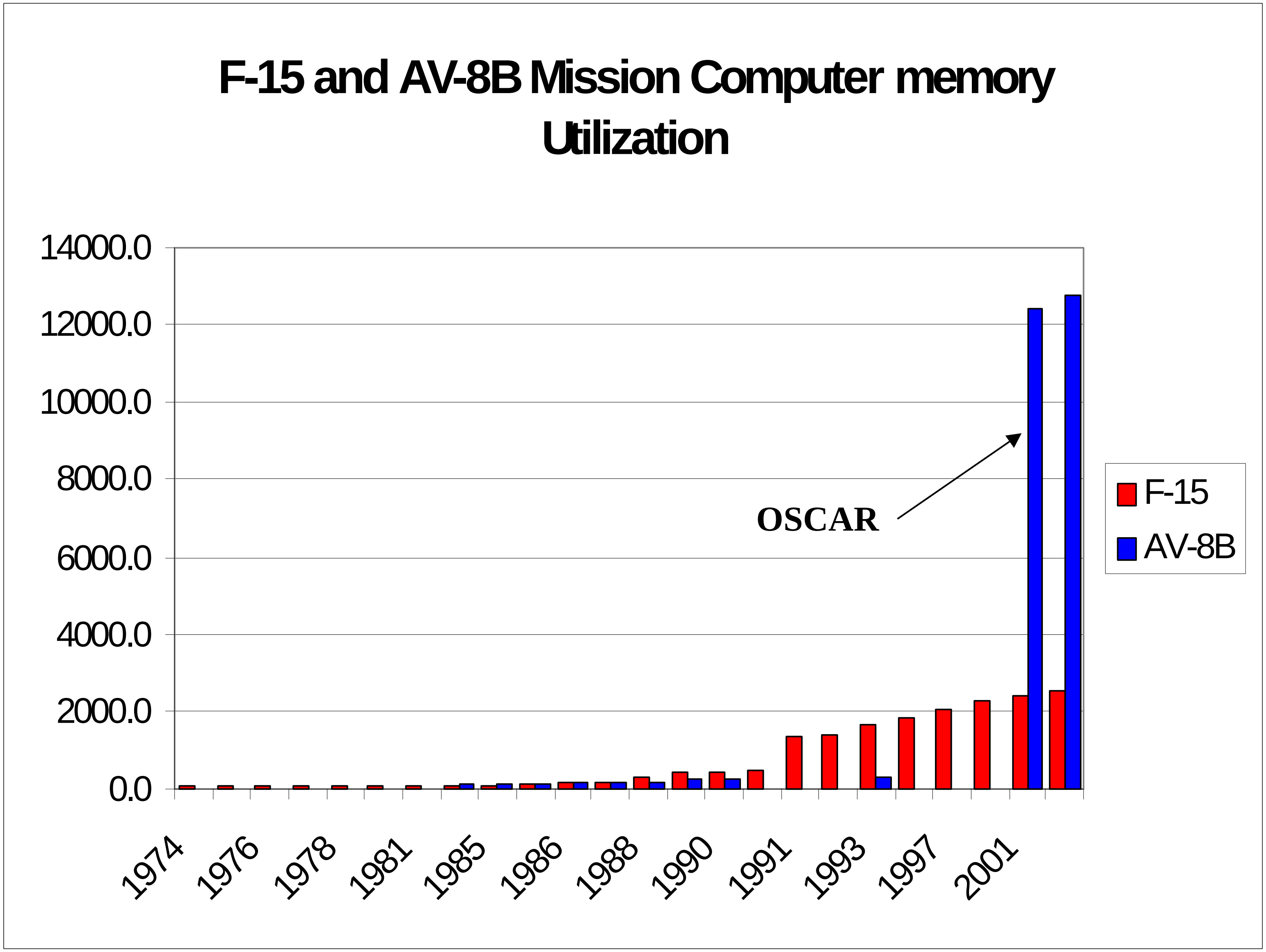
Memory and Throughput

- ***OOD is a big resource consumer.***
- ***The F-15 Central Computer OFP had already been converted from an assembly language to a HOL (Ada) in the early 1990's.***
- ***Felt comfortable with initial OSCAR estimates based on complexity of the F-15 aircraft versus the AV-8B, a six processor solution (on the F-15) versus a single processor, and the continued growth in available throughput in commercial processors.***

However, a 4x estimate turned into a 40x reality


F-15 Mission Computer Memory Utilization
3000.0
2500.0
2000.0
1500.0
1000.0
500.0
0.0
VHSIC Central Computer
•Ada language OFP
•Six processors with fault tolerance
F-15E
•Dual cockpit
•Full A/A and A/G capability
•Eight displays
1974
1976
1978
1981
1985
1986
1988
1990
1991
1993
1997
2001


F-15 and AV-8B Mission Computer (pre-OSCAR)
Memory Utilization
3000.0
2500.0
2000.0
1500.0
1000.0
500.0
0.0
1974
1976
1978
1981
1985
1986
1988
1990
1991
1993
1997
2001
F-15
AV-8B


F-15 and AV-8B Mission Computer memory Utilization
14000.0
12000.0
10000.0
8000.0
6000.0
4000.0
2000.0
0.0
1974
1976
1978
1981
1985
1986
1988
1990
1991
1993
1997
2001
OSCAR
F-15
AV-8B

Memory and Throughput Conclusions

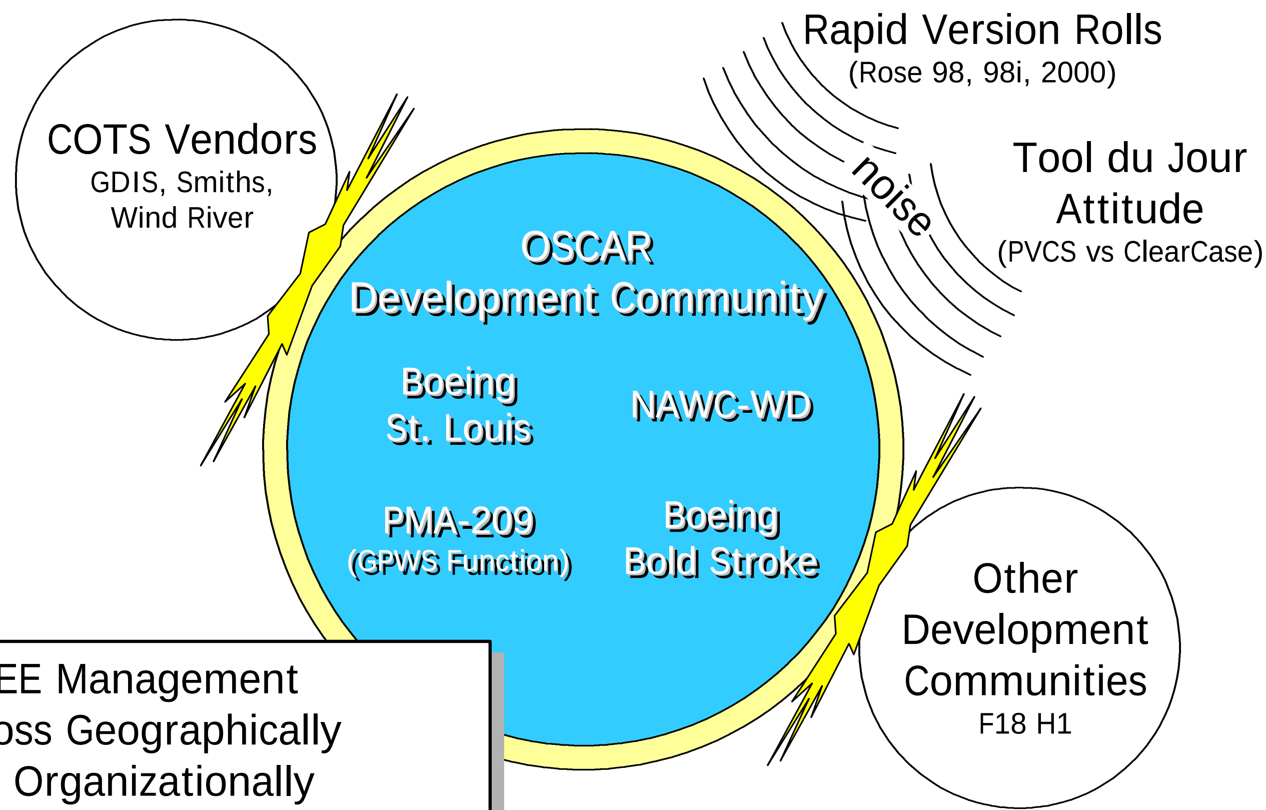
- ***Use of OOD has a tremendous impact on Memory usage.***
- ***Believe throughput impact is even greater, although more difficult to compare.***
- ***Lesson Learned - Use of OOD adds an order of magnitude (or more) to memory and throughput requirements.***

Tools Lessons

OSA Lessons Learned - Tools

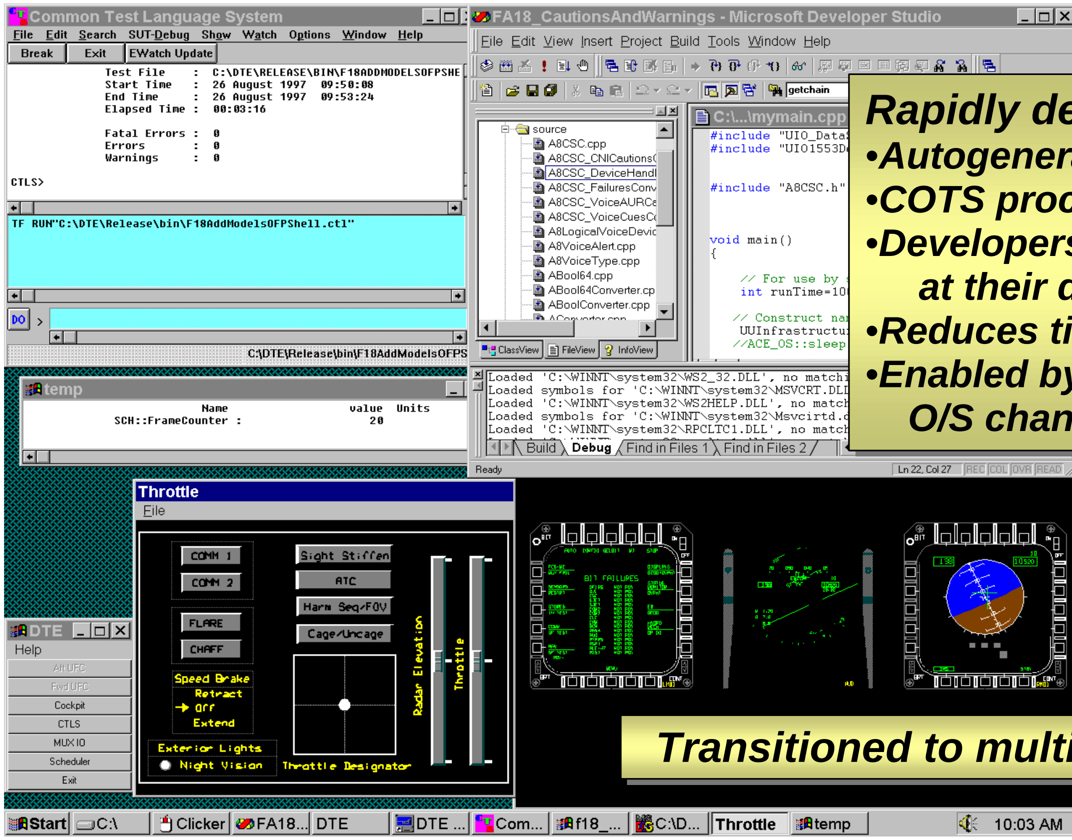
- ***Not All Commercial Tools Scale To Large Development Programs***
- ***Interoperability Of Commercial Tools Must Be Evaluated Prior To Selection***
- ***Keep Up With New Tool Versions To Maintain Vendor Support***
- ***Plan Tool Transitions***
- ***Utilize Dedicated Tool Engineers***

Tool Compatibility



S/SEE Management
Across Geographically
and Organizationally
Separated Development
Sites requires tight Tool CM

Desktop Test Environment



Rapidly design once
- ***Autogenerated code***
- ***COTS processors & tools***
- ***Developers run OFP at their desk***
- ***Reduces time and cost***
- ***Enabled by hardware and O/S change containment***

Transitioned to multiple programs

Summary

Lessons Learned Summary
(Most Critical)

- **COTS**
  - Use Existing Products
    - Don't Push Technology, Follow It (Cost/Schedule\Risk)
    - Use Technology Rolls To Satisfy Growth, Not Baseline Requirements
  - DOD Programs Have Limited Influence On Commercial Developments
    - Very-Very-Small Quantities Compared to Industry
  - COTS Does Well In Qualification Testing

- **Open Systems Design**
  - Cultivate/Develop Multiple Production Sources Up Front
  - Partition Software Workpackages Along Functional Lines (Self Contained Packages)

Lessons Learned Summary (Cont.)
(Most Critical)

- **C++ / OO Design**
  - Throughput Is Difficult To Estimate
  - Scale The Software To the EXISTING Computer Resources:
    - Memory, Throughput, I/O
  - In Order To Reuse Functional Software The Top Level Requirements **MUST** Be The Same
  - Reused Software Will Require Significant Rework
  - Process & Procedures Are No Substitute For A Stable, Well-Trained Workforce
  - Troubleshooting Transient Problems Is More Difficult in COTS Environment
  - Turnaround On Fixes Is Much Quicker
- **Functionality**
  - Document And Bound All Requirements
  - Limit New Functionality Until After Legacy Is Complete
  - Be Selective in Legacy Problem Fixing During Conversion
- **Use Multiple Metrics To Identify Problems**

Priority Order of the Top 10 OSCAR Lessons Learned

1 -- Document And Bound All Requirements

2 -- Reused Software Will Require Significant Rework

3 -- Process & Procedures Are No Substitute For A Stable Well Trained Workforce

4 -- Throughput Is Difficult To Estimate (OO)

5 -- Use Existing Products (COTS)

6 -- Use Multiple Metrics To Identify Problems

7 -- DOD Programs Have Limited Influence On Commercial Developments

8 -- Troubleshooting Transient Problems Is More Difficult

9 -- In Order To Reuse Functional Software The Top Level Requirements MUST Be The Same

10-- Partition Software Workpackages Along Functional Lines - (Self Contained Packages)

Summary

- ***How Are We Doing with Respect to Earlier Expectations?***
  - ***LCC savings and schedule improvements will not be realized until 2nd and 3rd upgrades***
  - ***Thruput estimates were off by an order of magnitude***
- ***Where Are We Going with the Open Systems Approach?***
  - ***Boeing Company roadmap for all legacy and future A/C system upgrades***
- ***Where Are We Going with Metrics Collection?***
  - ***Classes planned-vs-actuals is the best metric for program progress indicator***
  - ***Will continue to collect thru OC1.3 to set baseline***
- ***What Are We Going to "Do" with Lessons Learned Metrics?***
  - ***Compare to legacy systems metrics(where available) and produce / quantify data to establish baseline for F/A-18 & JSF systems development***
  - ***Incorporate lessons learned into Boeing-wide training programs***

The Next Step

Answer 5 Questions (Based On OSCAR Experiences)

1 -- How Fast Can The Investment Costs Be Recaptured?
2 -- Is OO/C++ Software Transparent To Hardware?
3 -- What is the Ratio Of New Functionality Development Costs Of OO/C++ vs. Assembly
4 -- Does OO/C++ Software Reduce Retest?
5 -- Is COTS Less Expensive?

The Next Steps - Develop A Plan

Develop A Plan/Process to Collect/Generate Data* that will Support the Determination of:

1 -- Actual Cost Of OSCAR Software Conversion

- *Use As Basis For Determining Investment Cost*
- *Factor Out New Functionality*
- *Requirements through Fleet Release*
- *Compare Against Original Estimates*
  - *If Different, Why?*

2 -- Actual Cost Of New Hardware (WMC / AMC)

- *Development Of Boxes*
  - *Use As Basis For Determining Investment Cost*
- *Unit Production Costs*
- *Compare Against Predictions*
- *Compare Against Dedicated Mil Spec. Box (Non-COTS)*

3 -- Was COTS Less Expensive?

- *Why or Why Not?*

****Note: Some Data May Not Be Available Until After The Completion Of OC1.1 & AMC&D***

The Next Steps - Develop A Plan

Develop A Plan/Process to Collect/Generate Data* that will Support the Determination of:

4 -- Actual Costs Of new Functionality

- *AMRAAM/13C (OC1.1)*
- *JDAM, HQ/SG (OC1.2)*

5 -- Comparsion With Assembly Language Version

- *Was It Cheaper to Develop? To Test?*
  - *Why?*

6 -- "Will OO & C++ Cause Less Retest In Subsequent OFPs?"

- *How?*
  - *Generate An OC1.2 Metric To Measure Unplanned Fixes To Legacy Caused By New Functionality*

7 -- Costs Associated With Migrating OSCAR OFP To New Processors

- *603e to 750*
- *750 to G4*
- *Was Hardware Transparent to Applications OFP?*
  - *If Not then Why?*
  - *Identify Issues*

****Note: Some Data May Not Be Available Until After The Completion Of OC1.1 & AMC&D***

The Next Steps - Determine the Pay Back

- ***Using***
  - ***The Initial Investment Costs***
  - ***Follow On New Development Costs***
- ***Determine***
  - ***How Much Software Must Be Written To Pay Back Initial Investment***

Bold Stroke
Open Systems Lessons Learned